Politecnico di Torino

Digital systems electronics
A.A. 2018/2019

Prof. G. Masera

Lab 06

14 may 2019

Berchialla Luca	236032
Laurasi Gjergji	238259
Mattei Andrea	233755
Lombardo Domenico Maria	233959

Introduction to the assigned problem

This relation deals with the design of a simple digital filter. From the given functional specifications, a final digital circuit has been implemented in VHDL including memories, control unit and data-path.
The design process will be analyzed using a bottom-up approach, starting from the individual components and their interconnections to the final custom designed FSM.

Overall specs description

As already mentioned, the final purpose of this activity is to implement a digital filter following the relation:

$$Y(n) = -0.5X(n) - 2X(n-1) + 4X(n-2) + 0.25X(n-3)$$

where $X(n)$ are the input stream data and $Y(n)$ the corresponding filtered data generated by the top equation.

The circuit starts by means of a $START$ signal which enable a loading process of the input data into a 1 kByte memory. Then, the circuit automatically filters the data stream exploiting the equation already provided and loads the output filtered data into a second memory. Finally, the circuit reports end of process asserting an $HIGH$ logical value to the $DONE$ signal and awaits until the next $START$.

Every used component will now be discussed as single blocks, and individually debugged:

Memories

As already discussed, the final implementation will save the input data in a 1kByte memory while memorizing the output data into an output memory.

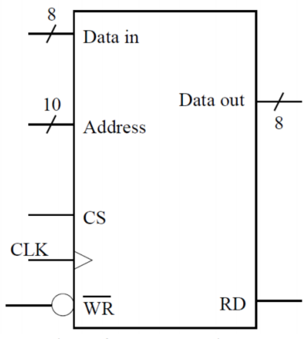

Figure 2 - Memory pinout

Figure 1: 1kByte memory

The memories store 1024 samples represented as 8 bit wide 2's complement values. The writing operation is synchronous with the positive edge of the clock, while the reading is asynchronous.

The samples must be stored in order, from address 0 to 1023.

Once the desired address is selected and the RD signal asserted, the output data are shown in the *dataout* parallel output.

The figure below shows the testbench for the register file.

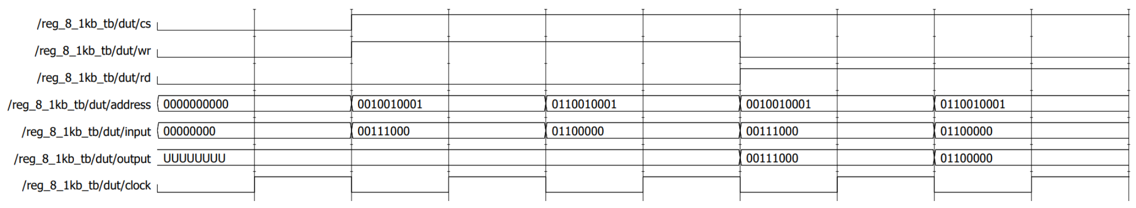

Figure 2: Memory testbench

Shift registers and registers

To be able to implement the already stated main equation, the circuit needs to perform 2s multiple multiplication and/or division.

The final implementation exploits the left and right shifting operations.

A 11 bit shift register has been implemented with parallel load and parallel output.

The *LOAD* signal must be asserted to load the *parallel input* data into the register, in a synchronous way with the positive edge of the clock.

The *SL* and *SR* commands permit the left or right shift of the internal memorized data.

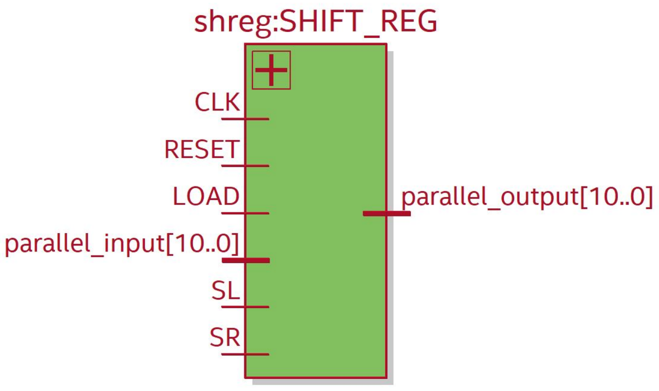

Figure 3: Shift register

The testbench for the 8-bit version is shown below:

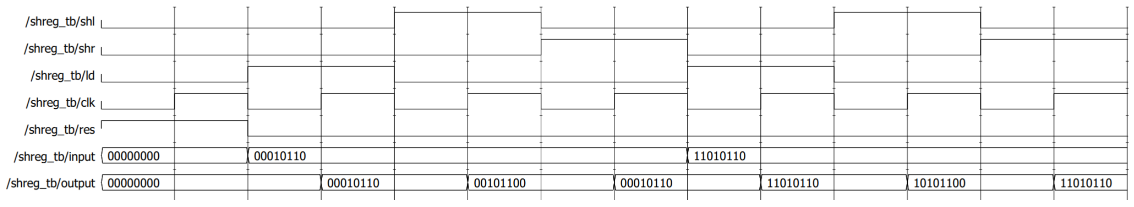

Figure 4: Shift register test bench

Adder

An 11-bit of a classical adder has been implemented, allowed to perform additions and subtractions by means of the *Cin* signal.

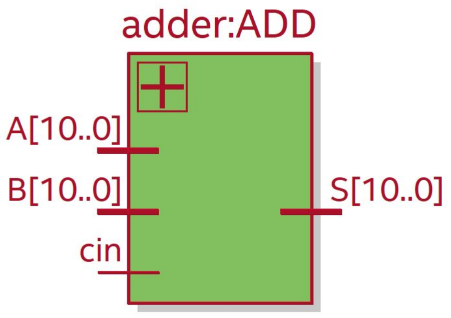

Figure 5: Adder

Below is shown the test bench of the designed adder.

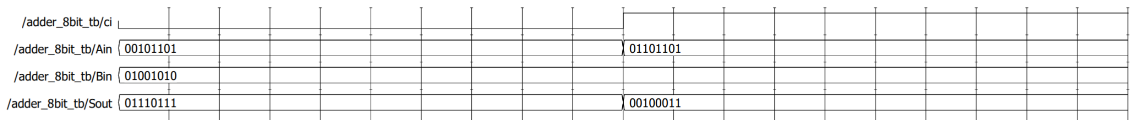

Figure 6: Adder test bench

Data converters

Finally, since the circuit must be able to determine both overflow and underflow it uses 2 blocks able to convert 8 bit signals to 11 bit and vice versa, as will be discussed in further sections.

The simulated test bench are shown below:

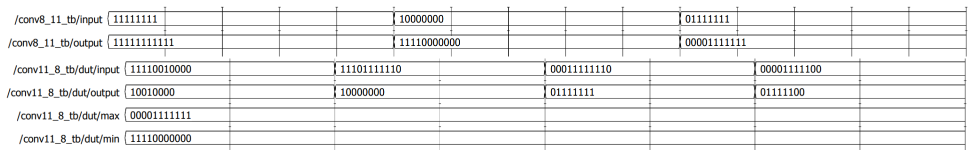

Figure 7: 8 to 11 bit converter; 11 to 8 bit converter

Counter

To be able to go through the various addresses an universal counter has also been implemented as shown: The counter can count up or down. Furthermore, the output

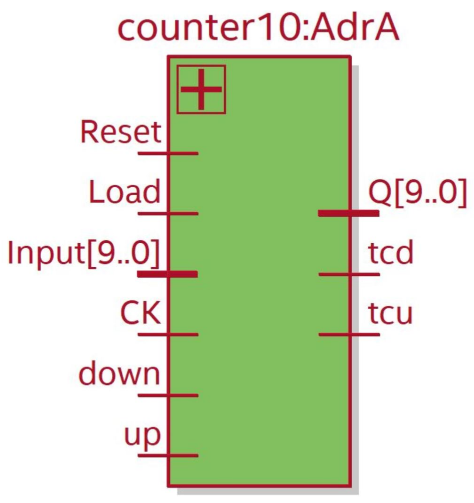

Figure 8: 10 bit counter

ports *tcd* and *tcu* have an high logical value when the counter reaches respectively its minimum or maximum possible value.

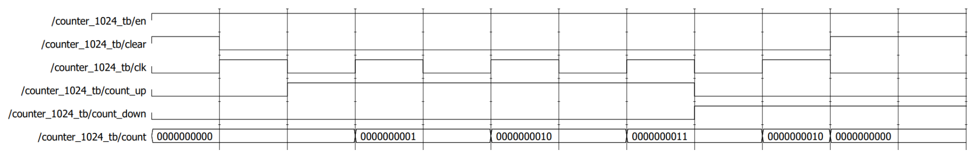

Figure 9: 10 bit counter test bench